Titus Lungu
11/12/2017

Udacity AIND, Foundations of AI: Project 3, Create a Domain-Independent Planner

	Breadth-First-Search		**Depth-First-Graph-Search**		**Uniform Cost Search**	
Air Cargo Problem 1	*Expansions*	43	*Expansions*	21	*Expansions*	55
	Goal Tests	56	*Goal Tests*	22	*Goal Tests*	57
	New Nodes	180	*New Nodes*	84	*New Nodes*	224
	Plan Length	6	*Plan Length*	20	*Plan Length*	6
	Time	0.032	*Time*	0.015	*Time*	0.038
Air Cargo Problem 2	*Expansions*	3343	*Expansions*	624	*Expansions*	4852
	Goal Tests	4609	*Goal Tests*	625	*Goal Tests*	4854
	New Nodes	30509	*New Nodes*	5602	*New Nodes*	44030
	Plan Length	9	*Plan Length*	619	*Plan Length*	9
	Time	14.324	*Time*	3.501	*Time*	12.068
Air Cargo Problem 3	*Expansions*	14663	*Expansions*	408	*Expansions*	18235
	Goal Tests	18098	*Goal Tests*	409	*Goal Tests*	18237
	New Nodes	129631	*New Nodes*	3364	*New Nodes*	159716
	Plan Length	12	*Plan Length*	392	*Plan Length*	12
	Time	105.178	*Time*	1.967	*Time*	53.048

	A* h_1		**A* h_ignore_preconditions**		**A* h_pg_levelsum**	
Air Cargo Problem 1	*Expansions*	55	*Expansions*	55	*Expansions*	11
	Goal Tests	57	*Goal Tests*	57	*Goal Tests*	13
	New Nodes	224	*New Nodes*	224	*New Nodes*	50
	Plan Length	6	*Plan Length*	6	*Plan Length*	6
	Time	0.037	*Time*	0.050	*Time*	0.593
Air Cargo Problem 2	*Expansions*	4852	*Expansions*	4852	*Expansions*	86
	Goal Tests	4854	*Goal Tests*	4854	*Goal Tests*	88
	New Nodes	44030	*New Nodes*	44030	*New Nodes*	841
	Plan Length	9	*Plan Length*	9	*Plan Length*	9
	Time	12.053	*Time*	13.292	*Time*	52.880
Air Cargo Problem 3	*Expansions*	18235	*Expansions*	18235	*Expansions*	318
	Goal Tests	18237	*Goal Tests*	18237	*Goal Tests*	320
	New Nodes	159716	*New Nodes*	159716	*New Nodes*	2934
	Plan Length	12	*Plan Length*	12	*Plan Length*	12
	Time	53.630	*Time*	58.056	*Time*	265.786

Depth first search consistently performs the most poorly, giving the largest plan length, though in the shortest amount of time. This is due to the fact that Depth first search must traverse the search tree all the way from top to bottom, reaching the goal state quickly but creating the least efficient plan. Uniform cost search, A* using h_1, and A* using h_ignore_preconditions return identical results, though ignore_preconditions takes the longest to create a plan. Breadth first search performs marginally faster for Air Cargo Problem 1, but otherwise A* h_1 and Uniform cost search are clearly the most efficient algorithms, and therefore deemed as providing the optimal solution. This is because Breadth first search traverses nodes sequentially while uniform cost search uses nodes of similar cost, and h_1 uses heuristics to efficiently solve the problem. Even though A* using levelsum consistently has the least expansions, goal tests, and new nodes, it's performance is slower than A* h_1 and Uniform cost search since it has to iterate through multiple levels to find the levelsum, depenidng on where the goals are. Below are the resulting plans for each problem, using the most optimal solution based on the recorded times and shortest plans from the tables above. As stated in module 32 (*A* Search 5*) of *Lesson 11: Search* of the video lessons, A* finds the lowest cost path if the heuristic underestimates the true cost, making h optimistic and admissible, supporting the results and analysis above. Similarly, in module 26 (*On Uniform Cost*) of the same lesson, uniform cost search is shown to search in the direction of the goal rather than in all directions, significantly improving results and time to find the goal.

Optimal solution for problem 1 using A* h_1
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Optimal solution for problem 2 using A* h_1
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Optimal solution for problem 3 using Uniform cost search
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)

Unload(C1, P1, JFK)
Unload(C2, P2, SFO)